МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №5

по дисциплине «Алгоритмы и структуры данных»

Тема: Хеш-таблицы

Вариант 24

Студент гр. 8304 Чешуин Д. И.

Преподаватель Фирсов М.А.

Санкт-Петербург

2019

Цель работы.

Познакомиться со структурой – хеш-таблица, способами её организации и обработки и устранения коллизий.

Постановка задачи.

1) Реализовать хеш-таблицу с цепочками, построение экземпляра таблицы на основе файла, а так же возможность поиска и удаления элемента по его ключу.

Описание алгоритма.

Программа считывает очередную строчку и разбирает её на слова, если первое из серии слов – s&d, то проводится поиск следующего за этим словом ключа и, при его наличии, удалении. Иначе, пара из двух слов добавляется в хеш-таблицу.

Добавление осуществляется с помощью вычисления хеша ключа и вставки пары элементов в конец списка, находящегося в ячейки таблицы с номером хеша. В случае, если количество элементов в таблице превышает её текущий размер, происходит её расширение и перевычисление всех ключей.

Спецификация программы.

Программа предназначена для управления хеш-таблицей.

Программа написана на языке C++. Входными данными являются слова. Они считываются из консоли или из файла. Выходными данными является история действий и состояний таблицы после каждой серии действий . Результат выводится в консоль либо в файл с именем – имя входного файла - result.

Тестирование.

Содержимое файла tests:

```
aaa bbb barell roll bring horizon
ccc ddd poets fall
eee fff rolling stone
10 asd banik soar
mmm 15
sea 150 150 mean
s&d aaa s&d bbb
s&d ddd s&d mmm
s&d banik s&d 10 s&d fff
```

Содержимое файла tests - result:

```
Pair - [aaa - bbb] - inserted.
Pair - [barell - roll] - inserted.
Pair - [bring - horizon] - inserted.
----------Table status!----------
Hash - 0
----[bring - horizon]
Hash - 1
----[aaa - bbb]
Hash - 2
Hash - 3
Hash - 4
Hash - 5
Hash - 6
----[barell - roll]
Hash - 7
Hash - 8
Hash - 9

------------------------------------------------

Pair - [ccc - ddd] - inserted.
Pair - [poets - fall] - inserted.
----------Table status!----------
Hash - 0
----[bring - horizon]
Hash - 1
----[aaa - bbb]
Hash - 2
Hash - 3
Hash - 4
Hash - 5
----[poets - fall]
Hash - 6
----[barell - roll]
Hash - 7
----[ccc - ddd]
Hash - 8
```

Hash - 9

Pair - [eee - fff] - inserted.
Pair - [rolling - stone] - inserted.
----------Table status!----------
Hash - 0
----[bring - horizon]
Hash - 1
----[aaa - bbb]
Hash - 2
Hash - 3
----[eee - fff]
Hash - 4
Hash - 5
----[poets - fall]
Hash - 6
----[barell - roll]
Hash - 7
----[ccc - ddd]
Hash - 8
Hash - 9
----[rolling - stone]

Pair - [10 - asd] - inserted.
Pair - [banik - soar] - inserted.
----------Table status!----------
Hash - 0
----[bring - horizon]
Hash - 1
----[aaa - bbb]
Hash - 2
Hash - 3
----[eee - fff]
Hash - 4
Hash - 5
----[poets - fall]
Hash - 6
----[barell - roll]
Hash - 7
----[ccc - ddd]
----[10 - asd]
----[banik - soar]
Hash - 8
Hash - 9
----[rolling - stone]

Pair - [mmm - 15] - inserted.
----------Table status!----------
Hash - 0
----[bring - horizon]
Hash - 1
----[aaa - bbb]
Hash - 2
Hash - 3
----[eee - fff]
Hash - 4
Hash - 5
----[poets - fall]
Hash - 6
----[barell - roll]
Hash - 7
----[ccc - ddd]
----[10 - asd]
----[banik - soar]
----[mmm - 15]
Hash - 8
Hash - 9
----[rolling - stone]

Pair - [sea - 150] - inserted.
Pair - [150 - mean] - inserted.
----------Table status!----------
Hash - 0
Hash - 1
Hash - 2
Hash - 3
----[eee - fff]
Hash - 4
Hash - 5
Hash - 6
----[barell - roll]
Hash - 7
----[mmm - 15]
Hash - 8
Hash - 9
Hash - 10
----[bring - horizon]
----[150 - mean]
Hash - 11
----[aaa - bbb]
Hash - 12
Hash - 13
----[sea - 150]
Hash - 14
Hash - 15
----[poets - fall]

Hash - 16
Hash - 17
----[ccc - ddd]
----[10 - asd]
----[banik - soar]
Hash - 18
Hash - 19
----[rolling - stone]

---

aaa - founded and removed
bbb - key not found.
----------Table status!----------
Hash - 0
Hash - 1
Hash - 2
Hash - 3
----[eee - fff]
Hash - 4
Hash - 5
Hash - 6
----[barell - roll]
Hash - 7
----[mmm - 15]
Hash - 8
Hash - 9
Hash - 10
----[bring - horizon]
----[150 - mean]
Hash - 11
Hash - 12
Hash - 13
----[sea - 150]
Hash - 14
Hash - 15
----[poets - fall]
Hash - 16
Hash - 17
----[ccc - ddd]
----[10 - asd]
----[banik - soar]
Hash - 18
Hash - 19
----[rolling - stone]

---

ddd - key not found.
mmm - founded and removed
----------Table status!----------
Hash - 0

Hash - 1
Hash - 2
Hash - 3
----[eee - fff]
Hash - 4
Hash - 5
Hash - 6
----[barell - roll]
Hash - 7
Hash - 8
Hash - 9
Hash - 10
----[bring - horizon]
----[150 - mean]
Hash - 11
Hash - 12
Hash - 13
----[sea - 150]
Hash - 14
Hash - 15
----[poets - fall]
Hash - 16
Hash - 17
----[ccc - ddd]
----[10 - asd]
----[banik - soar]
Hash - 18
Hash - 19
----[rolling - stone]

---

banik - founded and removed
10 - founded and removed
fff - key not found.
----------Table status!----------
Hash - 0
Hash - 1
Hash - 2
Hash - 3
----[eee - fff]
Hash - 4
Hash - 5
Hash - 6
----[barell - roll]
Hash - 7
Hash - 8
Hash - 9
Hash - 10
----[bring - horizon]
----[150 - mean]
Hash - 11

```
Hash - 12
Hash - 13
----[sea - 150]
Hash - 14
Hash - 15
----[poets - fall]
Hash - 16
Hash - 17
----[ccc - ddd]
Hash - 18
Hash - 19
----[rolling - stone]

---------------------------------------------
```

Анализ алгоритма.

Вставка, поиск и удаление происходит в среднем за константное время и за линейное время в худшем случае.

Описание функций и СД.

Методы класса MyHashTable

```
unsigned hash(const std::string& key) const;
```

Вычисляет хеш строки полиномиальным методом.

```
void insert(std::string key, ValueT value);
```

Вставляет пару элементов в хеш-таблицу.

```
ValueT get(const std::string& key) const;
```

Получает значение из хеш-таблицу по ключу.

```
void remove(const std::string& key);
```

Удаляет пару элементов по ключу.

```
bool search(const std::string& key) const;
```

Проверяет наличие ключа в таблице.

```
std::string toStr() const;
```

Возвращает строковое представление таблицы.

Вывод.

В ходе выполнения данной лабораторной работы реализован класс хеш-таблицы и методы работы с ним, изучен метод разрешения коллизий цепочками .

Приложение А. Исходный код программы.

myhashtable.h

```
#ifndef HASHTABLE_H
#define HASHTABLE_H

#include<list>
#include<iostream>
#include <sstream>

template<typename ValueT>
class HashTable
{
private:
    struct pair
    {
        std::string key_;
        ValueT value_;
    };

    int size_ = 10;
    int elementsCount_ = 0;
    std::list<pair>* array_ = nullptr;

    unsigned hash(const std::string& key) const;
    void resize(int size);

public:
    HashTable();
    ~HashTable();
    HashTable(const HashTable& table);
    void operator=(const HashTable& table);

    void insert(std::string key, ValueT value);
    ValueT get(const std::string& key) const;
    void remove(const std::string& key);
    bool search(const std::string& key) const;
    std::string toStr() const;
};

template<typename ValueT>
HashTable<ValueT>::HashTable()
{
    array_ = new std::list<pair>[size_];
}

template<typename ValueT>
HashTable<ValueT>::~HashTable()
{
```

```
    delete[] array_;
}

template<typename ValueT>
HashTable<ValueT>::HashTable(const HashTable& table)
{
    this->size_ = table.size_;
    for(int i = 0; i < size_; i++)
    {
        array_[i] = table.array_[i];
    }
}

template<typename ValueT>
void HashTable<ValueT>::operator=(const HashTable& table)
{
    this->size_ = table.size_;
    for(int i = 0; i < size_; i++)
    {
        array_[i] = table.array_[i];
    }
}

template<typename ValueT>
void HashTable<ValueT>::insert(std::string key, ValueT value)
{
    if(elementsCount_ >= size_)
    {
        resize(size_ * 2);
    }

    elementsCount_ += 1;

    unsigned hash_ = hash(key);

    pair newPair{key, value};
    array_[hash_].push_back(newPair);
}

template<typename ValueT>
ValueT HashTable<ValueT>::get(const std::string& key) const
{
     unsigned hash_ = hash(key);

     std::list<pair>& valueList = array_[hash_];

     for(auto iter = valueList.begin(); iter != valueList.end(); iter++)
     {
         if((*iter).key_ == key)
         {
             return (*iter).value_;
         }
     }
```

```
     std::cout << "Can't find key: " << key << std::endl;
     throw "Key doesn't exist";
}

template<typename ValueT>
void HashTable<ValueT>::remove(const std::string& key)
{
    elementsCount_ -= 1;
    unsigned hash_ = hash(key);

    std::list<pair>& valueList = array_[hash_];

    for(auto iter = valueList.begin(); iter != valueList.end(); iter++)
    {
        if((*iter).key_ == key)
        {
            valueList.erase(iter);

            return;
        }
    }

    std::cout << "Can't find key: " << key << std::endl;
    throw "Key doesn't exist";
}

template<typename ValueT>
bool HashTable<ValueT>::search(const std::string& key) const
{
    unsigned hash_ = hash(key);

    std::list<pair>& valueList = array_[hash_];

    for(auto iter = valueList.begin(); iter != valueList.end(); iter++)
    {
        if((*iter).key_ == key)
        {
            return true;
        }
    }

    return false;
}

template<typename ValueT>
unsigned HashTable<ValueT>::hash(const std::string& key) const
{
    unsigned hashKey = 0;

    for(unsigned i = 0; i < key.size(); i++)
    {
        hashKey += (hashKey << 8 ) + static_cast<unsigned>(key[i]);
```

```
        hashKey = hashKey % size_;
    }

    return hashKey;
}

template<typename ValueT>
void HashTable<ValueT>::resize(int size)
{
    int oldSize = size_;
    size_ = size;
    std::list<pair>* newArray = new std::list<pair>[size_];

    for(int i = 0; i < oldSize; i++)
    {
        for(auto iter = array_[i].begin(); iter != array_[i].end(); iter++)
        {
            unsigned hash_ = hash((*iter).key_);

            pair newPair{(*iter).key_, (*iter).value_};
            newArray[hash_].push_back(newPair);
        }
    }
    delete[] array_;
    array_ = newArray;
}

template<typename ValueT>
std::string HashTable<ValueT>::toStr() const
{
    std::stringstream out;
    for (int i = 0; i < size_; i++)
    {
        out << "Hash - " << i << std::endl;
        for(auto iter = array_[i].begin(); iter != array_[i].end(); iter++)
        {
            out << "----[" << (*iter).key_ << " - " + (*iter).value_ << "]" <<
std::endl;
        }
    }

    return out.str();
}
#endif // HASHTABLE_H
```

main.cpp

```
#include <iostream>
#include <string>
#include "ioManager.h"
#include <istream>
#include "myhashtable.h"
```

```
void executeComands(std::istream& input, IoManager& ioManager
,HashTable<std::string>& table);

int main(int argc, char** argv)
{
    HashTable<std::string> table;
    IoManager ioManager(argc, argv);

    std::istream* input = ioManager.nextStream();
    while(input != nullptr)
    {
        executeComands(*input, ioManager, table);

        delete input;
        input = ioManager.nextStream();
    }

    return 0;
}

void executeComands(std::istream& input, IoManager& ioManager,
HashTable<std::string>& table)
{
    std::stringstream buf;

    while(input.peek() != EOF)
    {
        std::string word;
        input >> word;

        if(word == "s&d")
        {
            input >> word;

            if(table.search(word))
            {
                table.remove(word);
                buf << word << " - founded and removed" << std::endl;
            }
            else
            {
                buf << word <<" - key not found." << std::endl;
            }
        }
        else
        {
            std::string value;

            input >> value;

            table.insert(word, value);
```

```
            buf << "Pair - [" << word << " - " << value << "] - inserted." <<
std::endl;
        }
    }

    buf << "-----------------Table status!-----------------" << std::endl;
    buf << table.toStr() << std::endl;
    buf << "-----------------------------------------------" << std::endl;

    ioManager.writeLine(buf.str());
}
```

ioManager.h

```
#ifndef CLIHANDLER_H
#define CLIHANDLER_H

#include <iostream>
#include <fstream>
#include <memory>
#include <sstream>

class IoManager
{
private:
    int argc_ = 0;
    char** argv_ = nullptr;
    int curArgNum_ = 1;

    std::istream* curInStream_ = nullptr;
    std::ostream* curOutStream_ = nullptr;

    void openNextStream();
public:
    typedef std::shared_ptr<IoManager> IoManagerP;

    IoManager(int argc, char** argv);
    ~IoManager();
    std::istream* nextStream();
    void writeLine(std::string line);
};

#endif // CLIHANDLER_H
```

ioManager.cpp

```
#include "ioManager.h"

IoManager::IoManager(int argc, char** argv)
{
    argc_ = argc;
    argv_ = argv;

    if(argc_ < 2)
    {
```

```
            curInStream_ = &std::cin;
            curOutStream_ = &std::cout;
        }
    }

    void IoManager::openNextStream()
    {
        if(curInStream_ == nullptr){
            curInStream_ = new std::ifstream();
            curOutStream_ = new std::ofstream();
        }

        if(curArgNum_ >= argc_)
        {
            if(curInStream_ != &std::cin)
            {
                delete curInStream_;
                delete curOutStream_;
            }

            curInStream_ = nullptr;
            curOutStream_ = nullptr;

            return;
        }

        std::ifstream* inFileStream =
static_cast<std::ifstream*>(curInStream_);
        if(inFileStream->is_open())
        {
            inFileStream->close();
        }
        std::ofstream* outFileStream =
static_cast<std::ofstream*>(curOutStream_);
        if(outFileStream->is_open())
        {
            outFileStream->close();
        }

        while(curArgNum_ < argc_ && !inFileStream->is_open())
        {
            std::cout << "Try to open file - ";
            std::cout << argv_[curArgNum_] << std::endl;

            inFileStream->open(argv_[curArgNum_]);

            if(inFileStream->is_open())
            {
                std::string outFile(argv_[curArgNum_]);
                outFile += " - results";
                outFileStream->open(outFile);

                std::cout << "File opened." << std::endl << std::endl;
```

```
                }
                else
                {
                    std::cout << "Can't open - file not founded" << std::endl <<
std::endl;
                }

                curArgNum_ += 1;
            }
        }

        std::istream* IoManager::nextStream()
        {
            if(curInStream_ == nullptr)
            {
                openNextStream();

                if(curInStream_ == nullptr)
                {
                    return nullptr;
                }
            }

            while(curInStream_->peek() == EOF)
            {
                openNextStream();

                if(curInStream_ == nullptr)
                {
                    return nullptr;
                }
            }

            std::string buffer;

            std::getline(*curInStream_, buffer);
            if(buffer == "")
            {
                return nullptr;
            }

            std::stringstream* sstream = new std::stringstream();
            *sstream << buffer;

            return sstream;
        }

        void IoManager::writeLine(std::string line)
        {
            *curOutStream_ << line << std::endl;
        }

        IoManager::~IoManager()
```

```
{
    if(curInStream_ != nullptr && curInStream_ != &std::cin)
    {
        delete  curInStream_;
        delete  curOutStream_;
    }
}
```